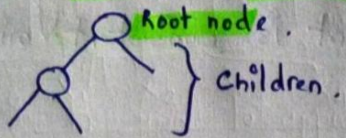
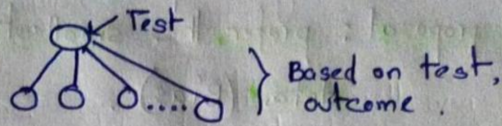
① **DECISION TREE (INTRODUCTION)** (Pages 1-8)

- Non linear function.
- A tree has nodes and branches.



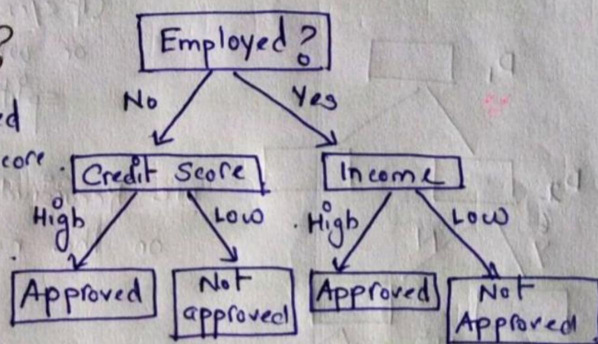


- 2 type of nodes - i) Decision Node. ii) Leaf node.
- In decision node we specify a choice or test based on this we can decide which direction we can go.
- The test is usually done on the value of a feature or attribute of the instance.
- Leaf node. indicate the classification of an example or value of the example.
- Decision tree can be used for classification and regression.
- Test will be done until we reach the value of the example / predicted value for classification, regression (target variable) or it can be probability.

Example - To give loan or not?

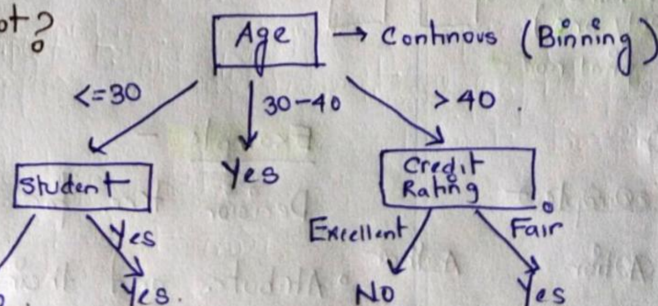
3 decision node - Employed
Credit Score.
Income.

4 leaf node - Approved x 2.
Not approved x 2.

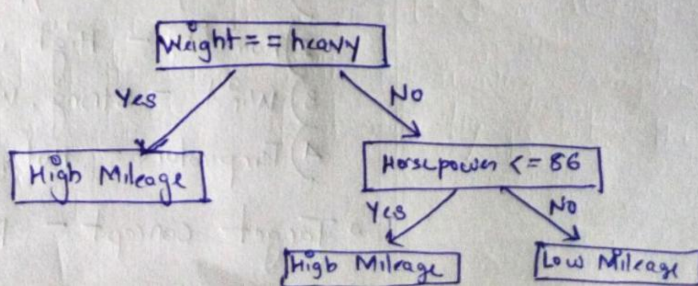


Will buy a computer or not?

Yes → Will buy
No → Not buy



Car Mileage prediction?



- By seeing a training set, it should build a decision tree.

Issues

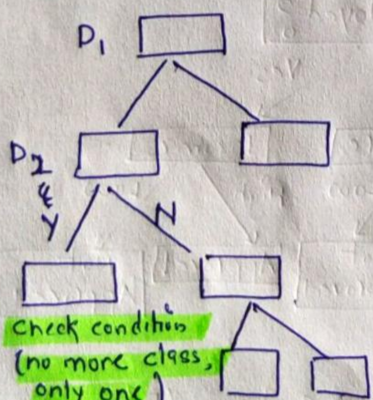
- Given some training examples, what decision tree should be generated?
- One proposal: prefer the smallest tree that is consistent with the data (Bias).

Possible Method:

- Search the space of decision trees for the smallest decision tree that fits the data.

Choose a tree which have less error.

- So choose bias.
- Once we choose decision tree as hypothesis space, so put some bias. Prefferably, we should have smaller trees (bias).
  Smaller tree means small number of nodes / small depth.
- Recursively built a decision tree - At every step, we check the condition (if yes we want to increase a tree, on which feature we want to split).



Check condition (no more class, only one) Then final node

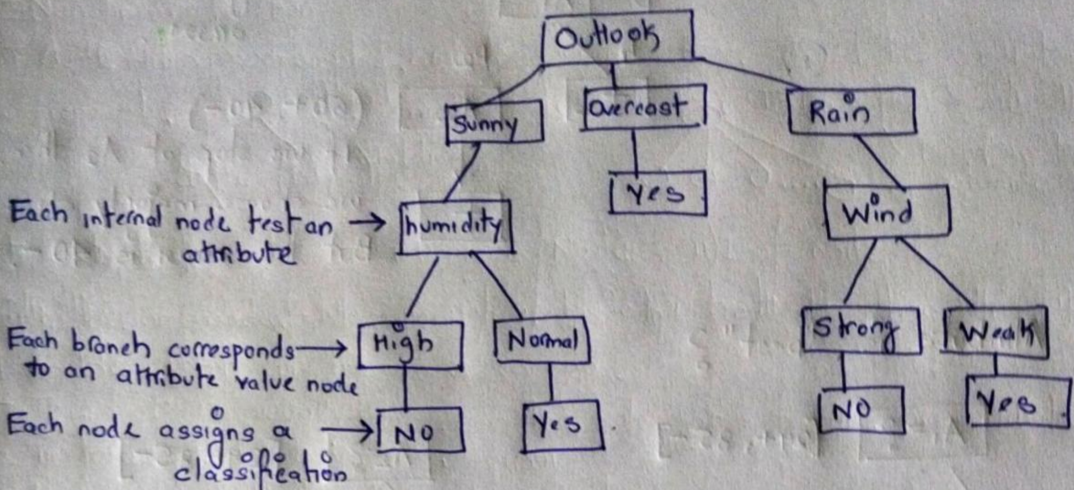
we Recursively build a decision tree based on the node.

Example -

Decision tree for Play Tennis

- Atributes and their values:
  1) Outlook - Sunny, Overcast, Rain
  2) Humidity - High, Normal
  3) Wind - Strong, Weak
  4) Temprature - Hot, Mild, cold.
- Target concept - Play tennis - Yes/No.

③ Sample decision tree for PlayTennis -



Question →	Outlook	Temperature	Humidity	Wind	PlayTennis
	Sunny	Hot	High	Weak	?
					(No)

Searching for a good tree ?

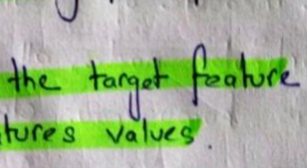
- The space of decision trees is too big for symmetric search.
- Stop and
  - return the a value of for the target feature or.
  - a distribution over target features values.
- Choose a test (eg input feature) to split on
  - For each value of test, build a subtree for those examples with this value for the test.

Two Main Questions - I) When to stop the decision tree ?
II) Which attribute to choose for split?

Learning Decision tree -

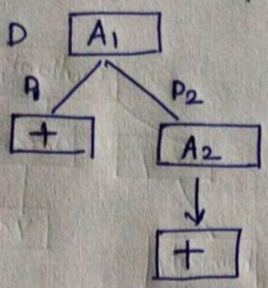
Atribute A= $A_1, A_2, A_3 \ldots, A_n$.

Decision → D.



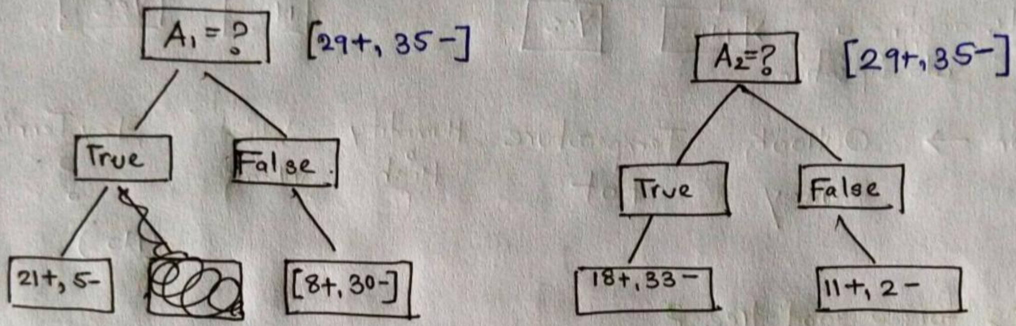
1) When to stop?

2) Which attribute to split on? → Split should give the smallest error.



(60+, 40−)
If we stop at A_2 then, we choose majority class
But the error is 40−.

→ Which Attribute is "best"?



Entropy − It is a measure of disorder in a system.
If in particular node, all value is of one class (either + or −) then it is homogenous class and entropy is 0, entropy is low.
- If it contain half-half class, then entropy is high.
- Leaf node have lowest entropy.

Principled criterion − Selection of an attribute to test at each node − choosing the most useful attribute for classifying examples.

Information gain − Measure how well a given attribute seperates the training example according to their target classification.
- This measure is used to select among the candidate attributes at each step while growing the tree.
- Gain is measure of how much we can reduce uncertainity (Values lies between 0,1).

5

- So if all examples have same target classification, information is high and information gain is high. (No certainity)
- In case of 50 - 50, information gain is low. (High Certainity)

So first choose entropy and then basis of that choose Information gain.

Entropy → High purity → Entropy = 0. $\boxed{\text{Entropy} = -\log_2 P}$ ↑ probability

- A measure for i) uncertainity ii) purity iii) Information gain.

- Information theory: Optimal length code assigns $(-\log_2 P)$ bits to message having probability P.

- S is the sample of training examples
  - P_+ is the proportion of positive example in S.
  - P_- is the proportion of negative example in S.

- Entropy(S) - Average optimal numbers of bits to encode information about certainity / uncertainity about S.

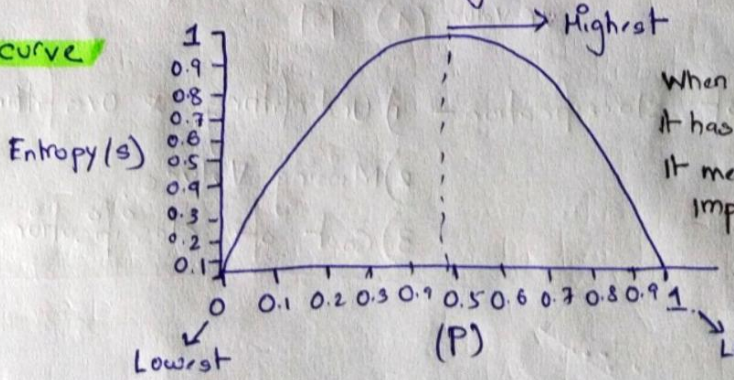
$$\text{Entropy}(S) = P_+(-\log_2 P_+) + P_-(-\log_2 P_-)$$

$$\boxed{\text{Entropy}(S) = -P_+ \log_2 P_+ - P_- \log_2 P_-}$$

$P_+=1, P_-=0 / P_+=0, P_+=1$, Entropy will be lowest, if class contain only one class

$P_+=\frac{1}{2}, P_-=\frac{1}{2}$, Entropy will be highest, if it contain equally both class

Entropy curve



When probability(P) = 0.5, it has highest entroy = 1. It means it has highest impurity.

P = 0.5 means equal probability (3 Yes, 3 NO)

- The entropy is 0 if the outcome is "certain".
- The entropy is maximum if we no knowledge of system (or any outcome is equally possible)

Information Gain -

Gain (S, A) - Expected reduction in entropy due to partioning S on attribute A.

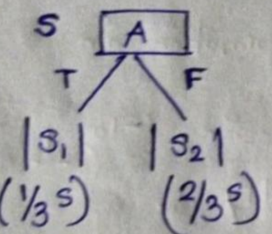
For every value of A, if we split the how many A.

$$Gain(S,A) = Entropy(S) - \sum_{v \in values(A)} |S_v| / |S| \; Entropy(S_v)$$

$$Entropy([29+, 35-]) = -29/64 \log_2 29/64 - 35/64 \log_2 35/64$$

$$= 0.99$$

Suppose,



$\Rightarrow$ Entropy $= \frac{1}{3}|S_1| + \frac{2}{3}|S_2|$

Information gain $= \underset{\text{(original Entropy)}}{Entropy(S)} - \underset{\text{(Resulting entropy)}}{\sum \frac{|S_v|}{|S_1|} Entropy(S_v)}$

- Information gain is high, on that split should be done. In other word, ~~reduce the entropga~~ reduction in entropy because we want low entropy and smaller decision tree.

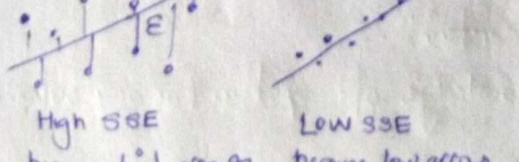
Other populare rule for splitting - Gini Index.

- Measure of node impurity

$$GINI_{node}(Node) = 1 - \sum_{c \in classes} [p(c)]^2$$

$$GINI_{split}(A) = \sum_{v \in values(A)} \frac{|S_v|}{|S|} GINI(N_v)$$

Practical Issues of classification -
1) Underfitting & Overfitting
2) Missing Values
3) Cost of classification.



High SSE because high error. Low R^2

Low SSE because low error. High R^2

Refer to Page 8